VoCo-LLaMA: Towards Vision Compression with Large Language Models

Supplementary Material

1. Additional Implement Details

1.1. Experiment Details

We elaborate on the training details and hyperparameter design of VoCo-LLaMA during the training process. Since our method only requires introducing VoCo tokens during the visual instruction tuning stage, the hyperparameters reported are specific to the fine-tuning stage. We followed [4] to configure the majority of the model parameters. Tab. 1 provides a detailed list of hyperparameters involved in the model training phase. Under the aforementioned parameter settings, VoCo-LLaMA is fine-tuned on 8 NVIDIA A100-SXM4-40GB GPUs.

Hyperparameter	Value
Overall batch size	128
Learning rate	2e-5
LR Scheduler	Cosine decay
DeepSpeed ZeRO Stage	ZeRO-3
Optimizer	AdamW
Warmup ratio	0.03
Epoch	2
Weight decay	0
Precision	bf16

Table 1. Hyperparameters of VoCo-LLaMA.

1.2. Instruction Tuning Details

In this subsection, we delve into the design of instructions and special tokens for VoCo-LLaMA during the instruction tuning stage, illustrating the details with concrete examples. As shown in Listing 1, by leveraging the pre-trained language model Vicuna [2] as the foundation for our method, we ensured a consistent system prompt and dialogue design to prevent the model from learning or adapting to new instruction formats during fine-tuning.

Furthermore, the VoCo-LLaMA architecture necessitates organizing the input sequence in a specific order, *i.e.*, vision tokens, VoCo tokens, and text tokens. However, demonstrated in Listing 1 [Common VLM], the standard "Conversation" and "Input" formats reveal that the special token <image> and its corresponding encoded <Vision Tokens> are situated in the middle of the dialogue sequence. To address this, we made modifications to align them in [VoCo-LLaMA]. We relocated the special token <image> to the forefront of the input prompt and appended the <VoCo Tokens> for compression. Concurrently, we refined the system prompt to accommodate the adjusted vision token position, thereby enabling the LLM to better comprehend and model the location of vision token.

1.3. Implementation Code

Listing 2 presents the concrete implementation of introducing VoCo tokens and the attention mask mechanism in VoCo-LLaMA, corresponding to Eq. (2) and (3) of our main manuscript. To facilitate reader understanding, we provide a brief code snippet implemented using PyTorch, accompanied by annotations.

2. Inference Efficiency Details

2.1. CUDA Time

We present a comparative efficiency analysis of VoCo-LLaMA, the baseline and full-caching strategies during the inference phase in Tab. 7 of our main manuscript. Here, we provide additional details on the CUDA time measurement during the inference phase. We primarily consider the following components that contribute to the reported CUDA time: image encoding time (if applicable), kv cache load time (if applicable), and transformers forward time. We exclude other computational times that are not dependent on the model itself and the caching strategy, such as model loading time, from the CUDA time measurement. For the baseline model, which is a regular visual language model, the inference process requires additional visual information encoding. Due to the lack of caching, the model must compute the transformer activations and generate text on uncompressed visual and textual tokens during inference.

Compared to the Full Caching strategy, the improvement in CUDA time achieved by VoCo-LLaMA can be attributed to two primary factors: Firstly, the time required to load the kv cache is significantly reduced, owing to the smaller storage size and shorter read time resulting from the 576× compression multiplier achieved by our approach. Secondly, the transformers forward time is also improved, although this is largely dependent on the length of the new input token (*i.e.,* text token), which is identical in both cases. However, VoCo-LLaMA's shorter prefix enables more efficient computation, leading to faster processing times. Notably, since both approaches cache transformer activations on top of vision tokens or compressed tokens, the image encoding duration is not a contributing factor to the overall time.

2.2. Storage Memory and FLOPs

The storage memory reported in Tab. 7 of our main manuscript corresponds to the full transformer activations

```
[Common VLM]
System prompt:
A chat between a curious user and an artificial intelligence assistant. The assistant gives helpful,
detailed, and polite answers to the user's questions.

Conversation:
USER: What is the man doing in the image?\n<image>
ASSISTANT: The man is sitting in a chair, possibly at his computer, with a bird (a parrot) on his
shoulder.
USER: Is there any other animal present in the image?
ASSISTANT: Yes, there is a cat in the image, which the man is petting while sitting at his computer.

Input:
A chat between a curious user and an artificial intelligence assistant. The assistant gives helpful,
detailed, and polite answers to the user's questions.
USER: What is the man doing in the image?\n<Vision Tokens>
ASSISTANT: The man is sitting in a chair, possibly at his computer, with a bird (a parrot) on his
shoulder.
USER: Is there any other animal present in the image?
ASSISTANT: Yes, there is a cat in the image, which the man is petting while sitting at his computer.
...

[VoCo-LLaMA]
System prompt:
<image>\n<VoCo Tokens>\n A chat between a curious user and an artificial intelligence assistant of the
given image. The assistant gives helpful, detailed, and polite answers to the user's questions.

Conversation:
USER: What is the man doing in the image?
ASSISTANT: The man is sitting in a chair, possibly at his computer, with a bird (a parrot) on his
shoulder.
USER: Is there any other animal present in the image?
ASSISTANT: Yes, there is a cat in the image, which the man is petting while sitting at his computer.

Input:
<Vision Tokens>\n<VoCo Tokens>\n A chat between a curious user and an artificial intelligence assistant
 of the given image. The assistant gives helpful, detailed, and polite answers to the user's questions.
USER: What is the man doing in the image?
ASSISTANT: The man is sitting in a chair, possibly at his computer, with a bird (a parrot) on his
shoulder.
USER: Is there any other animal present in the image?
ASSISTANT: Yes, there is a cat in the image, which the man is petting while sitting at his computer.
...
```

Listing 1. The instructions and special tokens details during the visual instruction tuning.

Method	Avg. Vision Token	Avg. Text Token	Total Testing Wall-time	1st Forward Time	2nd Forward Time
LLaVA-1.5 [4]	576	114	473.2 ms	-	-
LLaVA-1.5 + **VoCo**	576	114	366.3 ms	355.9 ms	10.4 ms

Table 2. The breakdown of the overall testing wall-time.

on top of the vision tokens or compressed VoCo tokens. This also refers to the size of the kv cache loaded during the inference process. The storage precision is bf16, which is consistent with the precision used in our training process.

2.3. Discussion About Two Forward Pass

2.3.1. Repeated Visual Requests

In many practical applications, it is common for multiple requests to involve the same image or video. In such cases, it is possible to cache the VoCo tokens obtained from the first request. Subsequently, for any new request related to the same image/video, these cached VoCo tokens can be

Method	Token	**VQA^{v2}**	**GQA**	**SEED**	**MME**	**MMB**	**POPE**
FastV [1]	64	55.0	46.1	44.8	807.9	48.0	48.0
VoCo-LLaMA	64	75.4	60.4	56.3	1397.2	60.5	82.1
FastV [1]	1	43.6	38.9	37.4	629.6	31.7	39.2
VoCo-LLaMA	1	72.3	57.0	53.7	1323.3	58.8	81.4

Table 3. Comparison with FastV [1] on common visual understanding benchmarks.

Method	LLM	Token	**VQA^{v2}**	**GQA**	**SEED^{I}**	**SQA^{I}**	**POPE**	**MME**	**MMB**	**TextVQA**	**Avg. Rate**
LLaVA-1.5-13B [4]	Vicuna-1.5-13B	576	80.0	63.3	68.2	71.6	85.9	1531.3	67.7	61.3	100%
LLaVA-1.5-13B + **VoCo**		1	74.5	59.1	63.4	70.5	82.5	1372.9	62.2	57.3	93.5%
LLaVA-1.6-7B [5]	Mistral-7B	2880	82.2	64.8	72.2	72.8	86.7	1498.0	68.7	65.7	100%
LLaVA-1.6-7B + **VoCo**		5	76.0	60.1	66.4	70.8	82.3	1346.9	62.4	61.0	92.7%
LLaVA-1.6-34B [5]	Hermes-Yi-34B	2880	83.7	67.1	75.9	81.8	87.7	1631.0	70.0	69.5	100%
LLaVA-1.6-34B + **VoCo**		5	78.9	62.8	71.2	81.6	84.3	1492.7	65.2	65.1	94.5%

Table 4. Results on more LLMs including Mistral-7B, Vicuna-1.5-13B, and Hermes-Yi-34B. The percentage values are the average compression retention rate mentioned in Sec. 4.2 of our paper. Higher average rate results from **excluding** the Lower-Bound model in calculations (i.e., VoCo / Upper Bound).

Method	Avg. Rate	Token	Training Wall-time	Training Memory
Q-Former [3]	57.2%	1	14,479 s	17.82 GB
VoCo-LLaMA	**83.7%**	1	18,672 s	19.34 GB

Table 5. Efficiency analysis of VoCo-LLaMA compared with Q-Former including training memory and wall-time.

reused, allowing for a single forward pass to complete the inference. This significantly reduces the inference cost.

2.3.2. Efficiency Analysis of Two Forward Pass

In this section, we discuss the inference efficiency of two forward passes without considering image caching. # represents the token count. Within the LLM, as the number of tokens increases linearly, the computational time for attention score computation grows quadratically with the number of input tokens, which dominates the time complexity. The time complexity of the two forward pass is O^2(#vision_tokens + #VoCo_tokens) + O^2(#VoCo_tokens + #text_tokens). The time complexity of upper bound method (LLaVA) is O^2(#vision_tokens + #text_tokens). Since #VoCo_tokens is far smaller than both #vision_tokens and #text_tokens, our method is more compute efficient. We provide the breakdown of the overall testing wall time in Tab. 2. However, when we cache images, we only need to consider the second forward time.

3. Additional Results

3.1. Visualized Results of Output

Fig. 1 illustrates the visualized examples of VoCo-LLaMA for vision compression. We compare our approach, which employs a single VoCo token, with the Upper Bound model described in Sec. 4.2 of our main manuscript (denoted as **U.B.** throughout the figure). Correct responses are highlighted in green, while incorrect responses are marked in red. We further show some of the failure cases. For questions that require attention to finer details of the input image, such as the third question in the second case, VoCo-LLaMA sometimes produces incorrect descriptions. When the VoCo token count increases to 8, these erroneous outputs are significantly improved, as the compression retention rate rises.

3.2. Relation to Concurrent Works

We elaborate that VoCo-LLaMA is the first method to use auto-regressive forward compression for large language models. However, recent contemporaneous approaches (e.g., FastV [1]) focus on compression and pruning of large vision language models. Our rationale stems from the distinction that "compression" involves generating new compressed tokens (e.g., Q-Former), while FastV "prunes" existing tokens without leveraging LLMs' auto-regressive compression to produce compressed tokens. VoCo-LLaMA is the first method that utilizes LLMs themselves for the generation of new compressed tokens.

Compared to training-free FastV, our method requires SFT, which incurs additional training overhead. Since the number of VoCo tokens introduced is much smaller than that of visual and text tokens, the training overhead is comparable to that of the Upper Bound model. Moreover, Tab. 3 shows that VoCo-LLaMA clearly outperforms FastV. Users can make a trade-off between higher performance and training-free requirement.

```python
import torch

# Example procedure for the VoCo-LLaMA training process.
class VoCoTrainingExample():
    def __init__(self, ImageEncoder, Tokenizer, Embedding):
        """
        ImageEncoder represents the visual encoder and subsequent linear layer.
        Tokenizer represents a textual tokeniser in the large language model.
        Embedding is used to transform the token after tokenize into embedding.
        """
        self.encode_images = ImageEncoder
        self.tokenizer_token = Tokenizer
        self.embed_tokens = Embedding

    def forward(prompt: str, voco_num: int, image: torch.Tensor)
        """
        Forward function in the main function.
        We show the core code of VoCo token introduced in Sec. 3.2 of our main manuscript.
        We simplify by replacing the 4D attention mask used in the actual implementation with a 2D attention mask.
        To make it easier for the reader to understand, we only briefly show the key parts.

        Args:
            prompt: a string containing the system prompt, question and other text inputs,
            (batch_size, L).
            voco_num: number of VoCo tokens inserted.
            image: a tensor of input image, (batch_size, 3, 336, 336).
        """

        # ...
        # Encode image into vision tokens, with the shape of (batch_size, 576, hidden_size)
        image_features = self.encode_images(images)

        # Insert VoCo tokens between vision tokens and text tokens.
        # The embedded prompt is with the shape of (batch_size, voco_num + L, hidden_size)
        prompt = "<VoCo>" * voco_num + prompt
        prompt_ids = self.tokenizer_token(prompt)
        prompt_embeds = self.embed_tokens(prompt_ids)

        # We save the index of VoCo token to calculate the attention mask.
        voco_idx = [image_features.shape[1] + i for i in range(voco_num)]

        # Corresponds to equation (2) in the paper.
        # The input embedding is arranged: [vision tokens, VoCo, text tokens] with the shape of (batch_size, 576 +
            voco_num + L, hidden_size)
        input_embeds = torch.cat((image_features, prompt_embeds), dim=1)

        # Constructing the standard causal attention mask in models of decoder-only.
        # More specifically, it is a lower triangular matrix.
        attention_mask = torch.full(input_embeds.shape[1], input_embeds.shape[1])
        attention_mask = torch.tril(attention_mask)

        # Constructing VoCo distillation attention mask, according to equation (3).
        # Due to the default casual mask setting, we only need to modify the part with the value of False.
        # i.e., keep the text token invisible to the vision token
        attention_mask[voco_idx[-1]+1:, :voco_idx[0]] = False
        # ...
```

Listing 2. Training of Python Pytorch code for matrix multiplication, based on llava.

3.3. Generalization Ability

Tab. 4 presents the results of more scales and LLMs, including Mistral-7B, Vicuna-1.5-13B, and Hermes-Yi-34B. VoCo-LLaMA demonstrates strong performance across various models, validating generality.

3.4. Discussion and Comparison with Q-Former

Tab. 5 compares training time and memory usage (per GPU) with Q-Former, using the same dataset, batch size, and hardware. Although Q-Former uses a lighter visual compression module, its limited compatibility with ZeRO-3 and suboptimal adaptation of BERT to Flash-attention result in minimal differences in training time and memory. Crucially, VoCo significantly outperforms Q-Former at same compression rate. The performance gap is not merely due to the more parameters but, more essentially, arises from our core idea of distilling visual tokens. This allows VoCo-LLaMA to maintain the original MLLM training paradigm, whereas Q-Former requires complicated training strategy and data due to its external modules.

4. Broader Impacts

Our research relates to the Large Language Models and Large Vision Language Models. Although the training data we employ is public and compliant, there are potential negative societal impacts to consider. One possible concern is the generation of unintended or malicious outputs, which could be used to spread disinformation. To mitigate these risks, we propose that future work focus on developing mechanisms for monitoring and detecting misuse, as well as implementing gated release strategies for models to ensure they are used responsibly.

5. Limitations

VoCo-LLaMA presents a novel solution for vision compression, but several limitations remain. Our training paradigm, while effective in enabling the model to comprehend compression tokens, inadvertently diminishes its ability to understand uncompressed vision tokens. Moreover, our model struggles to effectively handle tokens with diverse fine-grained compression levels. Addressing these limitations could unlock the door to infinite-length input visual-language models.

References

[1] Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models, 2024. 3

[2] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, 2023. 1

[3] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. BLIP-2: bootstrapping language-image pre-training with frozen image encoders and large language models. In *ICML*, 2023. 3

[4] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning, 2023. 1, 2, 3

[5] Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. Llava-next: Improved reasoning, ocr, and world knowledge, 2024. 3

User: What is the man in the suit doing in the image?

A: Standing with his arms crossed
B: Holding a cell phone
C: Taking a picture
D: Talking to someone

U.B. (576 tokens): A. Standing with his arms crossed.
VoCo-LLaMA (1 token): C. Taking a picture.

User: How many people are present in the image?

A: 1
B: 2
C: 3
D: 4

U.B. (576 tokens): B: 2.
VoCo-LLaMA (1 token): B: 2.

User: Where is the black writing located in the image?

A: On a white wall
B: On the man's suit
C: On the floor
D: On the woman's dress

U.B. (576 tokens): A: On a white wall.
VoCo-LLaMA (1 token): A: On a white wall.

User: What is the woman holding in hands in the image?

U.B. (576 tokens):
The woman is holding a handbag in the image.

VoCo-LLaMA (1 token): The woman is holding a purse in her hands in the image.

User: What is the main event taking place in the image?

A: A group of people playing soccer in a park
B: A crowd of people watching a live music performance
C: A group of people attending a political rally
D: A performer playing music while people pass by

U.B. (576 tokens):
B: A crowd of people watching a live music performance.
VoCo-LLaMA (1 token):
B: A crowd of people watching a live music performance.

User: Where is the audience looking in the image?

U.B. (576 tokens):
The audience is looking at the stage where the band is performing.

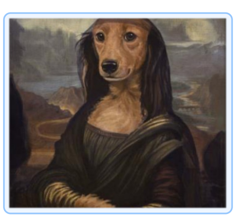
VoCo-LLaMA (1 token):
The audience is looking at a stage where a band is performing.

User: What’s happening in the scene?

U.B. (576 tokens):
The image depicts an outdoor event, possibly a festival or concert, with a large crowd of people gathered in front of a building. Several people are sitting in lawn chairs, while others are standing around, enjoying the live music. A man is playing a guitar on a stage, and a woman is singing into a microphone.There are multiple chairs scattered throughout the scene, with some people sitting in them and others standing nearby.

VoCo-LLaMA (1 token):
The image depicts a lively outdoor event where a man is performing on stage, playing the guitar and singing. A crowd of people, including at least 13 individuals, is gathered around to watch the performance. Some of the spectators are standing closer to the stage, while others are further back, enjoying the show from a distance.The setting appears to be a casual outdoor gathering, with a bench visible in the background. The audience members are engaged in the performance, creating a warm and inviting atmosphere.

VoCo-LLaMA (8 tokens):
The image depicts a lively outdoor event, possibly a concert or festival, with a large crowd of people gathered in front of a building. Many individuals are sitting in lawn chairs, spread across the scene, while others are standing and enjoying the event. There are several people playing musical instruments, with one person playing a guitar in the center of the scene. Another person is playing an instrument towards the right side of the image. In addition to the performers, there are various other attendees scattered throughout the scene, engaging in conversations and enjoying the atmosphere.



User: What is the painting in the image? Please describe it.

U.B. (576 tokens):
The painting in the image is a painting of a dog, specifically a dachshund, dressed in a Renaissance-style dress. The dog is wearing a long dress, a hat, and a necklace, giving it the appearance of a human figure. The painting is a unique and creative representation of a dog in a historical context.

VoCo-LLaMA (1 token):
The painting in the image is a parody of a famous painting, featuring a dog in a human-like pose. The dog is wearing a tie and appears to be sitting on a chair, giving the impression of a human figure. The painting is a humorous and creative take on the original artwork.

Figure 1. Examples of instructions and model outputs demonstrating the vision compression capabilities of VoCo-LLaMA. Green and red represent the correct and incorrect responses, respectively.